HelioScan: a software framework for controlling in vivo microscopy setups with high hardware flexibility, functional diversity and extendibility

Langer, D ; van 't Hoff, M ; Keller, A J ; Nagaraja, C ; Pfäffli, O A ; Göldi, M ; Kasper, H ; Helmchen, F

Abstract: Intravital microscopy such as in vivo imaging of brain dynamics is often performed with custom-built microscope setups controlled by custom-written software to meet specific requirements. Continuous technological advancement in the field has created a need for new control software that is flexible enough to support the biological researcher with innovative imaging techniques and provide the developer with a solid platform for quickly and easily implementing new extensions. Here, we introduce HelioScan, a software package written in LabVIEW, as a platform serving this dual role. HelioScan is designed as a collection of components that can be flexibly assembled into microscope control software tailored to the particular hardware and functionality requirements. Moreover, HelioScan provides a software framework, within which new functionality can be implemented in a quick and structured manner. A specific HelioScan application assembles at run-time from individual software components, based on user-definable configuration files. Due to its component-based architecture, HelioScan can exploit synergies of multiple developers working in parallel on different components in a community effort. We exemplify the capabilities and versatility of HelioScan by demonstrating several in vivo brain imaging modes, including camera-based intrinsic optical signal imaging for functional mapping of cortical areas, standard two-photon laser-scanning microscopy using galvanometric mirrors, and high-speed in vivo two-photon calcium imaging using either acousto-optic deflectors or a resonant scanner. We recommend HelioScan as a convenient software framework for the in vivo imaging community.

DOI: https://doi.org/10.1016/j.jneumeth.2013.02.006

Journal of Neuroscience Methods 215 (2013) 38–52

Contents lists available at SciVerse ScienceDirect

Journal of Neuroscience Methods

journal homepage: www.elsevier.com/locate/jneumeth

Computational Neuroscience

HelioScan: A software framework for controlling in vivo microscopy setups with high hardware flexibility, functional diversity and extendibility

Dominik Langer[a,b,1], Marcel van 't Hoff[a,2], Andreas J. Keller[b,c], Chetan Nagaraja[d], Oliver A. Pfäffli[a], Maurice Göldi[a,3], Hansjörg Kasper[a], Fritjof Helmchen[a,b,*]

[a] Brain Research Institute, University of Zurich, Winterthurerstrasse 190, CH-8057 Zurich, Switzerland
[b] Neuroscience Center Zurich, University of Zurich and ETH Zurich, Winterthurerstrasse 190, CH-8057 Zurich, Switzerland
[c] Institute of Neuroinformatics, University of Zurich and ETH Zurich, Winterthurerstrasse 190, CH-8057 Zurich, Switzerland
[d] Department of Neuroscience, Uppsala University, Uppsala, Sweden

HIGHLIGHTS

- HelioScan, a software framework for vivo microscopy control applications, written in the LabVIEW environment, is introduced.
- HelioScan application assembles at run-time from configurable software components and provides high flexibility and easy extendibility.
- We present four use cases demonstrating HelioScan's capabilities, in particular with regard to recent trends in the field of in vivo calcium imaging.
- A variety of imaging modalities are supported (video camera, galvanometric scan mirrors, acousto-optic deflectors, resonant scanners).
- We recommend HelioScan as a convenient software framework for the in vivo imaging community.

ARTICLE INFO

Article history:
Received 6 December 2012
Received in revised form 5 February 2013
Accepted 6 February 2013

Keywords:
Two-photon laser scanning microscopy
Intrinsic optical imaging
Control software
LabVIEW

ABSTRACT

Intravital microscopy such as in vivo imaging of brain dynamics is often performed with custom-built microscope setups controlled by custom-written software to meet specific requirements. Continuous technological advancement in the field has created a need for new control software that is flexible enough to support the biological researcher with innovative imaging techniques and provide the developer with a solid platform for quickly and easily implementing new extensions. Here, we introduce HelioScan, a software package written in LabVIEW, as a platform serving this dual role. HelioScan is designed as a collection of components that can be flexibly assembled into microscope control software tailored to the particular hardware and functionality requirements. Moreover, HelioScan provides a software framework, within which new functionality can be implemented in a quick and structured manner. A specific HelioScan application assembles at run-time from individual software components, based on user-definable configuration files. Due to its component-based architecture, HelioScan can exploit synergies of multiple developers working in parallel on different components in a community effort. We exemplify the capabilities and versatility of HelioScan by demonstrating several in vivo brain imaging modes, including camera-based intrinsic optical signal imaging for functional mapping of cortical areas, standard two-photon laser-scanning microscopy using galvanometric mirrors, and high-speed in vivo two-photon calcium imaging using either acousto-optic deflectors or a resonant scanner. We recommend HelioScan as a convenient software framework for the in vivo imaging community.

1. Introduction

Intravital optical imaging techniques are widely used to study tissue functions in various biomedical research fields (Pittet and Weissleder, 2011). In particular, two-photon laser-scanning microscopy (Denk et al., 1990) has revolutionized research in neuroscience, immunology, and other fields by enabling high-resolution imaging of cellular structure and function in intact tissue in the living animal (for review see Helmchen and Denk, 2005; Lütcke and Helmchen, 2011; Wilt et al., 2009). Using in vivo two-photon calcium imaging, for example, one can now characterize

* Corresponding author at: Brain Research Institute, University of Zurich, Winterthurerstrasse 190, CH-8057 Zurich, Switzerland. Tel.: +41 44 635 3340; fax: +41 44 635 3303.
E-mail address: helmchen@hifo.uzh.ch (F. Helmchen).
[1] Present address: Solution Providers Schweiz AG, Neugutstrasse 89, CH-8600 Dübendorf, Switzerland.
[2] Present address: European Laboratory for Non-Linear Spectroscopy, Via Nella Carrara 1, I-50019 Sesto Fiorentino, Italy.
[3] Present address: Artificial Intelligence Laboratory, Department of Informatics, University of Zurich, Andreasstrasse 15, CH-8050 Zurich, Switzerland.

0165-0270/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.jneumeth.2013.02.006

sensory-evoked and behavior-related activity in local neuronal populations of rodent neocortex (e.g., Dombeck et al., 2007; Harvey et al., 2012; Huber et al., 2012; Kerr et al., 2005; Margolis et al., 2012; Ohki et al., 2005; Stosiek et al., 2003). Different from microscopic analysis of fixed histological tissue sections, in vivo microscopy often requires additional equipment (e.g., for animal monitoring or stimulation) and specially tailored dynamic imaging modes. To meet such specific needs, as well as for budget reasons (Stuurman et al., 2007), microscope setups for in vivo imaging are often custom-built and controlled by custom-written software. Whereas customized microscope setups have obvious immediate advantages such as their high flexibility, there are a number of points to be considered, especially for the design of custom control software.

Often, a custom-made imaging setup is a unique combination of hardware components. Among the reasons for microscope diversity are (1) different functional requirements; (2) historical reasons (i.e., when individual setups are built sequentially, the state of the art for individual hardware components usually changes over time); (3) personal preferences of the involved engineers or scientists; (4) budget constraints; and (5) the still rapid pace at which entirely new imaging technologies and modalities emerge. For two-photon microscopy, this hardware diversity is reflected for example in diverse microscope stages, various signal generation and acquisition devices, and distinct types of laser scanning devices (Fig. 1A) (Engelbrecht et al., 2008; Göbel and Helmchen, 2007b; Grewe et al., 2010; Kirkby et al., 2010; Nguyen et al., 2001). Optimally, the control software is able to deal with this diversity of hardware components.

Functionality of an imaging system is desired to be tailored to specific research objectives. These objectives may differ even between researchers from the same laboratory and even more so across different laboratories. Furthermore, they are evolving over time. Hence, a general consensus of what an imaging system should be capable of accomplishing can hardly be reached (Chi, 2008). A prominent example of different kinds of functionalities are different special scanning trajectories in the case of two-photon microscope setups, such as three-dimensional spirals (Göbel et al., 2007), tilted frames (Göbel and Helmchen, 2007b), arbitrary lines (Lillis et al., 2008), or distributed points with sub-patterns for random access scanning (Duemani Reddy et al., 2008; Grewe et al., 2011; Otsu et al., 2008; Ranganathan and Koester, 2010; Salome et al., 2006) (Fig. 1B). Likewise, various types of stimulations that are typically used in neuroscience experiments need to be controlled (Fig. 1C). Control software should either provide a wide spectrum of functionality or it should be easily extendible to incorporate the newest advancements in methods development.

In the academic environment, software devised to control an experimental setup is often not only used by the original programmer, but also by their colleagues or successors. Typically, little attention is paid to usability. Specific problems revealed by follow-up users may include unlabelled graphical user interface (GUI) elements, run-time errors occurring when the software user interface is accessed in a different manner (sequence, speed, etc.) compared to the original developer, as well as poor or non-existing documentation. Custom software also often lacks defined structure and documentation, making it difficult for follow-up programmers to build upon existing code. In addition, when monolithic software is modified independently by different researchers wanting to adapt it to their individual needs, different versions quickly arise, which can further branch into sub-versions, etc. New functionality introduced in one version then cannot be used by another researcher using another version. Also, a programmer is often not aware that a certain problem has already been solved by a colleague. Thus, a clear demand exists for wider software usability for larger groups of users as well as for a team of multiple programmers.

A software framework for custom microscope setups that considers these issues (hardware diversity, functional diversity, and

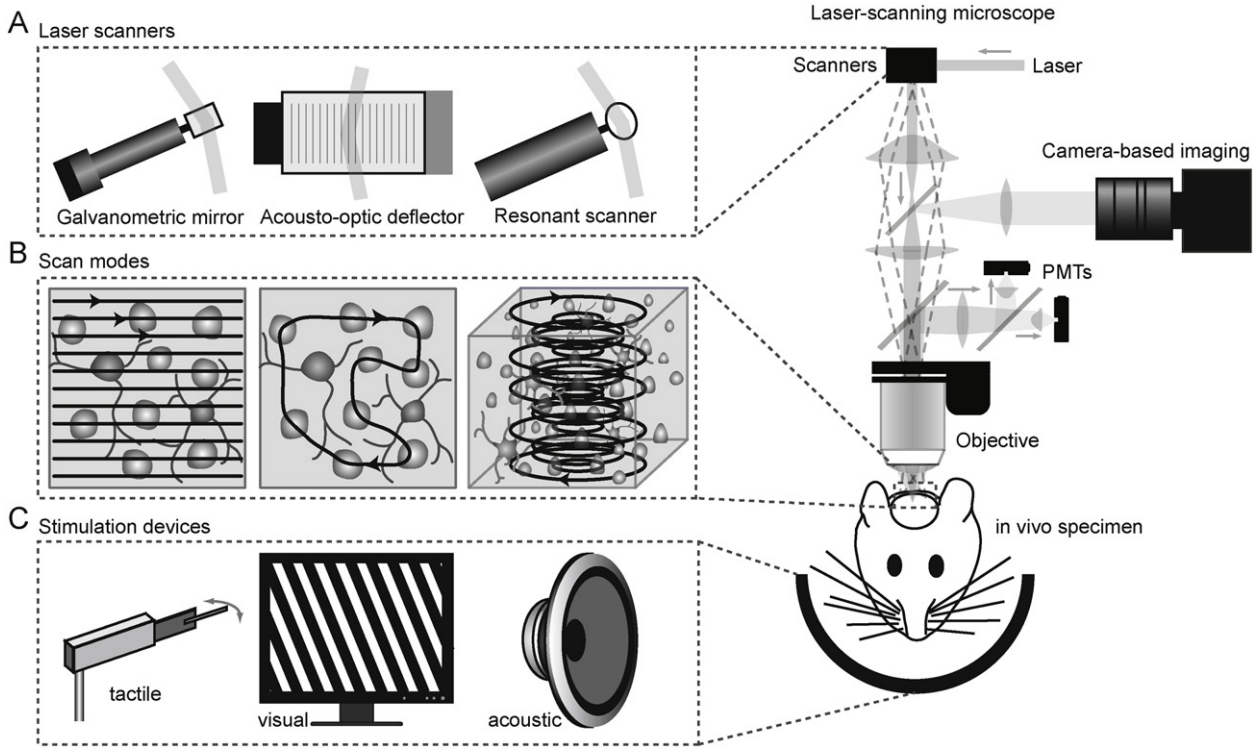

Fig. 1. Diversity of hardware components and functionality encountered in two-photon brain imaging experiments. (A) Different types of laser scanners used to deflect the laser beam in the microscope. (B) Depending on the scanner type, different types of laser scan trajectories may be suitable to meet the experimental objectives. (C) Depending on the brain area under investigation, different types of stimulation devices are required to probe the neuronal populations imaged.

usability) is highly desirable, especially in the expanding field of in vivo two-photon microscopy. Although several software packages are available – either commercial (Chi, 2008) or non-commercial (Hartell, 2007; Nguyen et al., 2009; Pologruto et al., 2003) – these do not provide flexible frameworks but rather ready-made packages. Here, our goal was to design and implement a microscopy software framework that has (1) the flexibility to support many different hardware and functional combinations, (2) good extendibility, which requires understandable, documented code that can be edited by multiple developers in parallel without interference, and (3) high usability. HelioScan serves as such a software framework facilitating the development of microscope control applications following a structured approach. The collection of components built inside this framework can be used to assemble a microscope control application with high flexibility. These two aspects complement each other. First, from the developer's perspective, one does not have to start implementing an application from scratch but can make use of many already existing components. Second, from the user's perspective, many needs can easily be met by appropriately assembling and configuring existing components. If a piece is missing, it is only this very piece that has to be newly implemented, with the framework providing helpful guidance.

We first describe the concept of HelioScan and its main architectural components. Subsequently, we demonstrate its capabilities in four use cases of different in vivo imaging modalities.

2. Results

2.1. HelioScan concept

A HelioScan application assembles at run-time from individual software components. Accordingly, the requirements set in the introduction have been implemented as follows.

To promote *flexibility toward diversity*, distinct hardware components are encapsulated as software components. Different hardware combinations can be addressed by the corresponding combination of software components. Likewise, different units of functionality are abstracted as distinct software components. Thus, a given functionality of a certain type can be swapped for another functionality of the same type. If a software component supporting a specific hardware device or functionality does not yet exist, it can be created by deriving it from the corresponding abstract component type.

The existing components, especially those defining the actual framework, are well-documented, well-structured and programmed according to established style-rules (Blume, 2007). Adhering to standard rules both ensures *understandibility* of existing code and promotes *extendibility*. In order to develop a new component, the developer does not have to understand all existing components. Rather, due to the hierarchical architecture of the framework, he will only have to understand in a top-down fashion the vertical slice to which his new component will belong to.

Multiple developers can work independently on their own component classes without interfering with each other. Since components should be as self-contained as possible, the developer can test most of their functionality independent from the main program. A version control system avoids version branching and ensures that a single central development branch persists over time.

Usability is mainly determined by the usability of the individual components and is thus in the responsibility of the individual developer. For a component to be part of the official HelioScan distribution, it should comply with basic quality standards such as listed in (Blume, 2007). Components with low usability can still be implemented, but their poor quality remains restricted and does not affect the remaining part of HelioScan.

2.2. Framework architecture

HelioScan features a main virtual instrument (VI), which provides the environment for components to assemble (in LabVIEW, a VI corresponds to a procedure in text-based procedural programming languages). HelioScan components can be classified into a number of predefined component types. Components of the same component type are – in principle at least – interchangeable. Following an object-oriented approach, components are realized as LabVIEW classes. All classes of the same component type share an abstract base class and hence a common interface of member functions through which they can be accessed and manipulated by other program code (Fig. 2A). As a consequence, each component instance can be swapped at run-time for an instance of another class of the same component type. As a common rule, whenever we identified a functional or physical entity that could occur in different forms depending on the physical hardware of microscopes (e.g., motorized stages) or functional requirements of the user (e.g., scan trajectories), we specified these peculiarities as different child classes of a common abstract base class that defines the corresponding generic component type with its typical interfaces (Fig. 2A and C). In cases where the interaction between two components required the call of methods not present at the level of the generic interface inherited from their component-type-defining abstract base class, we used so-called adapter classes containing a single method handling the particular interaction. Typically, a component instance loads a configuration file at start-up. Among various configurable parameters, components read in which other components to load. Hence, component-specific configuration files control the self-assembly process of HelioScan at run-time.

We distinguish between top-level components (TLCs) and subcomponents. Subcomponents serve merely as attributes of superordinate components and there are no fundamental restrictions in the number of instances allowed for each type of subcomponent. For each type of TLC, only a single instance exists at a time, which can be directly accessed by the HelioScan main VI. For each TLC type, the HelioScan main VI provides a subpanel into which the TLC can load its own GUI (Fig. 3). TLCs can interact with each other as well as with the HelioScan main VI in several ways. First, each TLC class can in principle determine the particular classes used for the other types of TLCs (Fig. 2B). Second, TLCs can send trigger messages to each other. Each TLC contains a central *run* method with a state machine that reacts to incoming triggers with state transitions (Fig. 4A). We distinguish between common triggers that are compatible with all TLC component types and TLC-type-specific triggers that can be sent to and can be understood by only a specific type of TLC. Third, a TLC can execute public member functions of other TLCs (Fig. 2C). For this purpose, the calling TLC can temporarily lock the called TLC using a mutex (mutual exclusion) mechanism to avoid race conditions (Fig. 4A). Fourth, each TLC can register as an observer of any other TLC type and thus receive copies of the data produced by the observed component. This allows implementing arbitrary, even branched data processing pipelines.

2.3. Types of top-level components (TLCs)

In the following, we briefly introduce the different types of TLCs (Fig. 2B). In addition to the purpose of each component type, we will also describe some typical interactions between them.

The central TLC is the *ImagingMode*, which specifies the mode of image acquisition, such as image capture with a camera or frame-by-frame laser scanning. The ImagingMode class being instanced

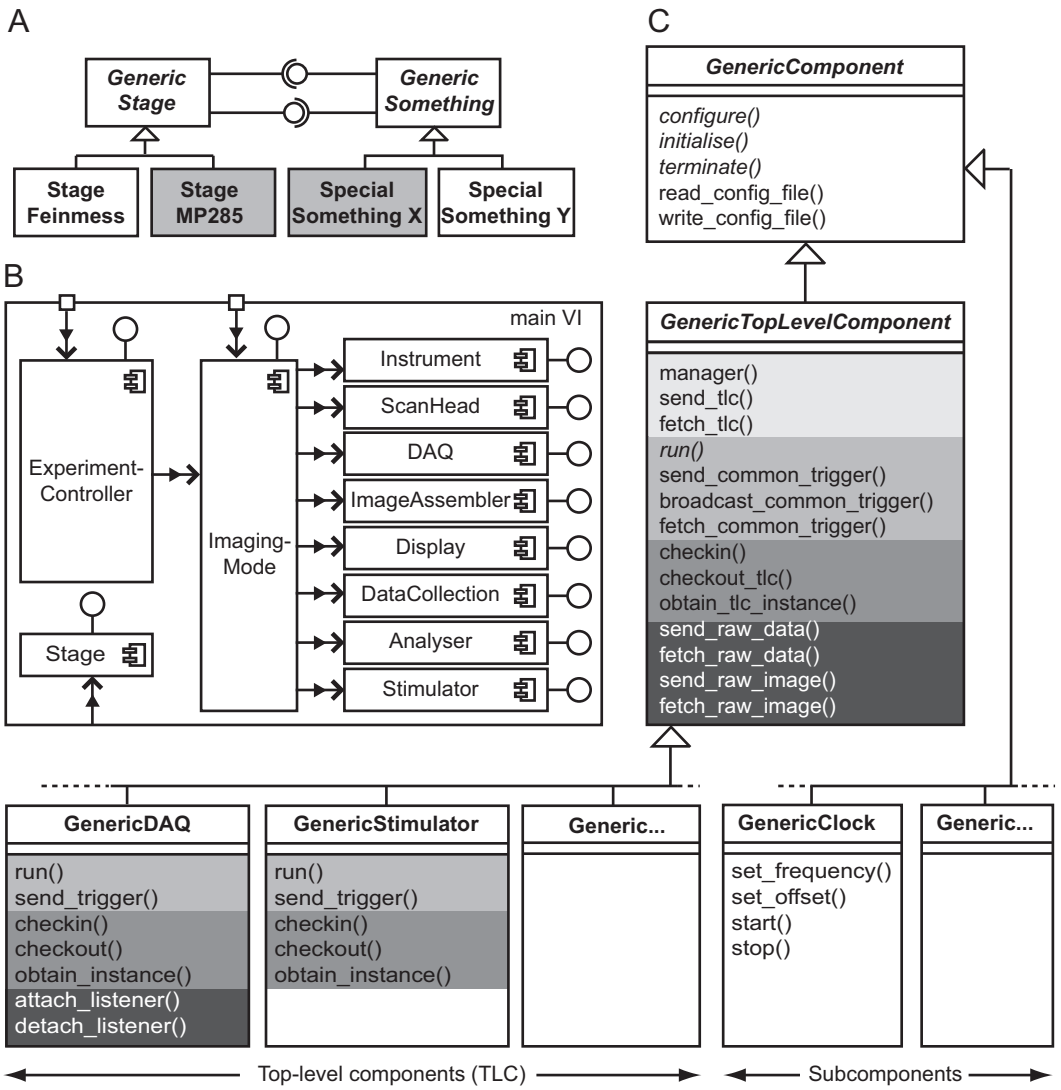

Fig. 2. Architecture of the HelioScan software framework. (A) Different implementations of a particular type of top-level component (TLC) can interact with different implementations of another type of TLC (e.g., gray with gray) via the interface inherited from their respective abstract base classes (here: GenericStage and GenericSomething, respectively). Arrows: class inheritance; lollipops: interfaces, via which TLCs can interact with each other. (B) Schematic of how TLCs are instanced and dispatched in a running HelioScan application. The ImagingMode (ImgMd) is either directly selected by the user or dispatched by the ExperimentController (ExpCtr, directly selected by the user). The ImagingMode, based on its configuration file, loads and dispatches all other types of TLCs, except for the Stage component, which is loaded and dispatched by the main VI directly. (C) Hierarchy of HelioScan component classes. GenericComponent is the common parent, from which all other components are derived. In contrast to subcomponents, TLCs have GenericTopLevelComponent as a common intermediate base class. For each component class, some selected interface methods are displayed (marked by different gray levels in the case of TLCs): methods related to the TLC queue (light gray); methods related to the state machine (medium gray); mutex-related methods (dark gray); methods related to the raw data and raw image queues (darkest gray).

is usually selected by the user on the main VI's GUI (Fig. 3D). Via its own GUI, the ImagingMode can accept user input to refine the respective mode of image acquisition (e.g., the dimensions in pixels, scan rate, or number of frames to acquire).The ImagingMode usually loads and dispatches most of the TLCs of the component types listed below based on its configuration file (Fig. 2B). In laser-scanning modes, the ImagingMode pre-configures a Trajectory sub-component based on user input and sends it to the ScanHead TLC.

The *ScanHead* component controls the scanners that move the focal point of a laser through the sample in laser-scanning imaging modes. The ScanHead receives a pre-configured Trajectory object from the ImagingMode, which defines the path of the laser focal point through the sample. While the basic shape of this path is determined already when the ScanHead receives the Trajectory, modification of some trajectory properties such as scaling factor, rotation angle or laser intensity might still be possible during scanning, depending on the actual ScanHead and Trajectory classes used. Currently, we distinguish two kinds of Trajectory, both requiring their specialized type of ScanHead. The first kind of so-called 'voxel trajectories' calculate the scan coordinates on the host PC, while the corresponding ScanHead converts these coordinates to command signals for the individual scanners. The second kind of Trajectory, so-called 'FPGA trajectories', directly or indirectly hand over their parameters to an FPGA module, which computes the scan coordinates on the fly (i.e., the command signal value for each trajectory pixel position is calculated on the FPGA when scanned).

The *DAQ* component is responsible for continuous data acquisition, such as reading from an analog-to-digital converter (ADC) that digitizes PMT signals. Thus, the DAQ component acts as a source of data and typically stands at the beginning of a data processing pipeline as mentioned above. Other TLCs that want to make use of this data can register as observers and will receive copies of the data blocks. The DAQ can read in data and pass it on to its observers

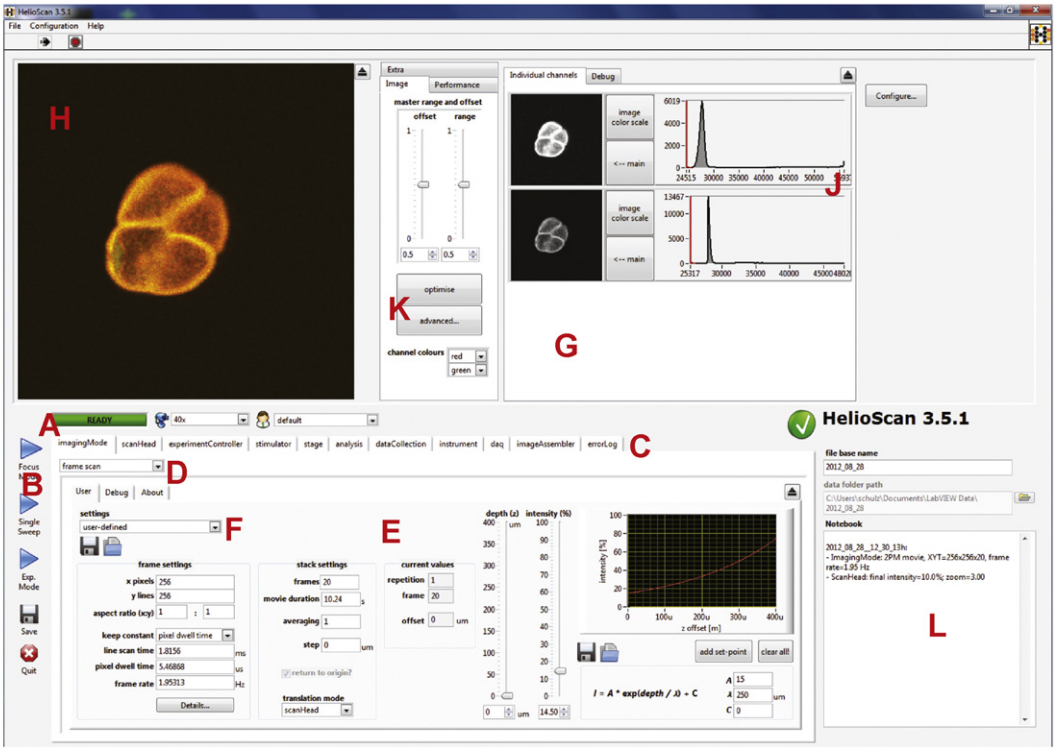

Fig. 3. HelioScan's main GUI. (A) Status bar indicating the progress of loading the top-level components (TLCs), as well as the readiness of the application. (B) Start buttons for the different run modes (i.e., free-run mode, single sweep mode and experiment mode). (C) Tabs harboring the GUIs of the individual TLCs. (D) Drop-down menu to select the ImagingMode component to use. (E) GUI of the currently selected ImagingMode component. (F) Drop-down menu allowing to load previously saved settings values of the ImagingMode user interface. (G) GUI of the currently loaded Display component. (H) Main image display with merged channels. The channels to merge and their colors can be selected by the user. (J) Individual channels with histogram of pixel values. (K) Controls to automatically or manually adjust range and offset of the individual channels. (L) Notebook on which a description of each saved file is automatically inserted. The user can enter additional information.

either as a stream of samples from an arbitrary number of channels, or as complete images in the case of camera-based devices.

The *ImageAssembler* usually registers as an observer at the DAQ and assembles the raw pixel stream into complete images. Alternatively, when the DAQ component encapsulates a device that produces images instead of a pixel stream, an ImageAssembler can still be involved in order to perform post-processing of these raw images.

The *DataCollection* stores image data to disk for later retrieval. In addition, it can provide via its GUI the functionality for the user to access already acquired image data. The DataCollection usually obtains image data from the ImageAssembler, at which the DataCollection has registered as an observer. If the ImageAssembler acts as a bottleneck at very high data acquisition rates, one can also use an architecture in which a DataCollection component registers as a DAQ observer and streams raw image data (be it a pixel stream or raw images from a camera) directly to disk.

The *Display* is responsible for displaying images to the user. Acting as a data sink, it attaches to at least one source of images, typically to the ImageAssembler and the DataCollection. Since it receives the image as copies, it can carry out arbitrary image manipulation prior to display without affecting the original data.

The *Analyser* can perform on-line or off-line analysis on image data received from components with which it has registered as an observer. These are typically the ImageAssembler and the DataCollection, but a scenario where data is obtained directly from the DAQ component is also possible.

A *Stimulator* component can encapsulate and control any kind of device capable of generating stimuli delivered to the specimen.

The *Instrument* can control further parts of the microscope not covered by any other TLC, such as detectors or light sources.

The *Stage* component is responsible for controlling and reading out a motorized microscope stage. Because the stage is usually given for a particular setup and does not depend on the mode of image acquisition, it is usually defined by the main VI itself, rather than by any of the TLCs. Like other TLCs, the Stage component uses dynamic polymorphism (i.e., the call of different methods depending on the actual component classes loaded) to control different hardware (Fig. 4).

Finally, the *ExperimentController* can carry out complex or looped measurement protocols involving repeated image acquisition. A HelioScan user starts image acquisition in one of three run modes. In 'focus mode', images are acquired continuously until acquisition is explicitly stopped by the user. They are not stored by the DataCollection component. This mode is well-suited to find a particular region of interest or adjust illumination power under visual feedback. In 'single sweep mode', a predefined number of images is acquired, with the option to save them to disk. In 'experiment mode', the ExperimentController component takes control over image acquisition. Depending on the specific ExperimentController class chosen as well as its configuration, repeated acquisition sweeps may be triggered by a TTL signal or time periods. Individual sweeps may even be configured to include sequences of different imaging modes or parameter settings.

In the following, we will present four use cases employing different image acquisition systems, namely a video camera and three different types of laser scanners combined with fluorescence detection on a photomultiplier. For all four cases we present example in vivo imaging data from mouse neocortex. Exploiting the modular architecture of HelioScan, several programmers independently implemented the various imaging modes available for these four modalities.

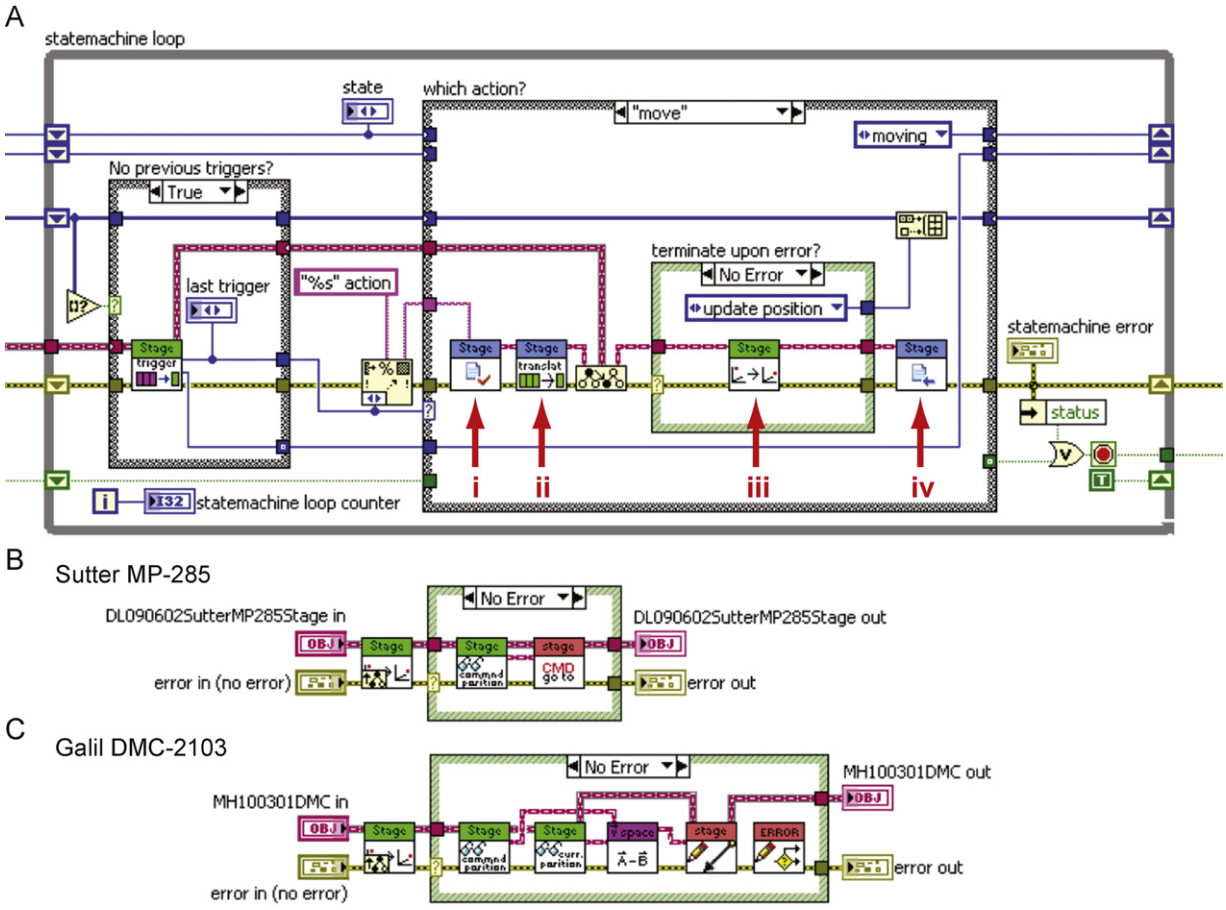

Fig. 4. Dynamic polymorphism at the TLC level. (A) State machine loop of a *Stage* scaffold class compatible with multiple hardware-specific child classes. When a *move* message is obtained via the message queue, (i) the mutex functionality of the Stage TLC obtains exclusive write access to the Stage instance, (ii) a three-dimensional vector is fetched from the move command queue, (iii) the *move* method of the currently instanced Stage class is called using late binding, and (iv) the mutex releases the lock on the Stage instance. (B) The *move* method of the Stage class *StageDL090602SutterMP285* communicates with the Sutter MP-285 stage controller using proprietary commands sent via a RS-232 interface. (C) Likewise, the *move* method of the Stage class *StageMH100301DMC* uses another set of commands to communicate with the Galil DMC-2103 stage controller.

2.4. Use case 1: Intrinsic optical signal imaging with a video camera

Intrinsic optical signal (IOS) imaging refers to the measurement of spatiotemporal changes in light reflectance of brain tissue that correlate with local changes in neural activity (Frostig et al., 1990). When shining light onto an exposed brain or through a thinned skull, intensity changes of the reflected light can be detected with a sensitive monochrome video camera during sensory stimulation, providing information about the localization of brain areas prominently involved in neural processing of the provided stimulus (Grinvald et al., 2005). In our setup, the IOS imaging system is integrated into a custom-built two-photon laser-scanning microscope (Fig. 5A). An LED ring mounted directly on the microscope objective provides illumination at different wavelengths. For IOS imaging, the dichroic mirror steering two-photon excited fluorescence light into the detector system is temporarily removed, and another mirror reflecting the IOS image onto the camera is blocking the laser beam path.

HelioScan provides two camera imaging modes needed for IOS imaging experiments (Fig. 5A and B). The first one is a simple video camera mode that enables the acquisition of reference images of the brain's surface vasculature under green LED light illumination for optimal contrast. The resulting blood vessel map is used to register the IOS against visual landmarks (Fig. 5D). The second imaging mode is used to acquire the actual IOS (Fig. 5C), typically under red LED light illumination (Fig. 5A). Importantly, as the IOS is very small (typically on the order of 0.1% reflectance change), it is important to operate the camera near saturation. As an example, we applied the two camera-based imaging modes to localize the cortical representations of individual whiskers in the mouse barrel cortex (Fig. 5C–E). Post-mortem staining for cytochrome C oxidase (COX) (Land and Simons, 1985) and image registration using blood vessel landmarks validated the imaging results, confirming that single-whisker stimulation evoked IOS spots correspond to the position of the respective individual barrel column (Fig. 5E). Beyond enabling functional mapping, the IOS imaging mode allows to record time-series of images, which can be used to analyze the time-course of the IOS (Fig. 5F).

2.5. Use case 2: Two-photon imaging with galvanometric mirrors

Galvanometric mirrors have been traditionally used for laser beam deflection in laser-scanning microscopy and are still the most common scanner type in confocal and two-photon microscopes. Their deflection angle can be controlled either by analog or digital input to the controller electronics. Whereas galvanometric mirrors are easy to control, their acceleration and frequency response are limited by mechanical inertia. In this and the subsequent use cases, we exemplify imaging modes with in vivo calcium measurements of neuronal activity in mouse neocortex (Kerr et al., 2005; Stosiek et al., 2003). Neurons were loaded with the green

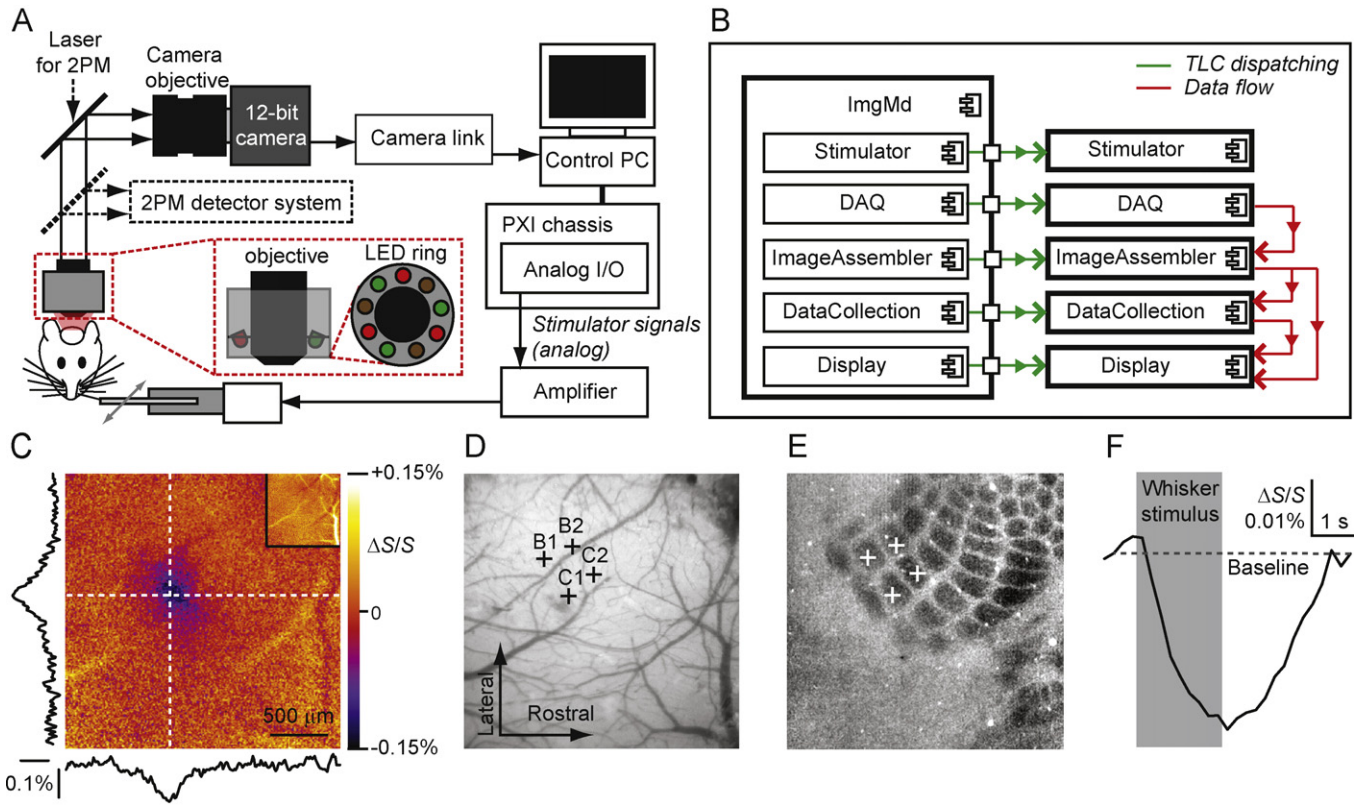

Fig. 5. Intrinsic optical signal imaging. (A) Hardware wiring diagram for an IOS imaging system integrated into a two-photon microscope setup. (B) Interaction of the most important HelioScan components. The DAQ encapsulates the camera and sends the acquired images to the ImageAssembler for spatial and temporal binning. Binned images are forwarded to the Display and to the DataCollection. The DataCollection can replay images upon user input by sending them to the Display. Video camera mode (used to obtain subfigure D) and intrinsic imaging mode (used to obtain subfigures C and F) differ only in the ImagingMode and DataCollection classes used. (C) IOS imaging in the mouse barrel cortex. Time-averaged IOS upon C1 whisker stimulation. (D) Position of four cortical whisker representations determined with IOS imaging (same experiment as in C). (E) COX staining of the cortical tissue, confirming the barrel positions shown D. (F) Time-course of the IOS (average of 12 trials, separate experiment). Stimulus application is indicated in gray.

fluorescent calcium indicator dye Oregon Green BAPTA-1 (OGB-1) using the standard bulk-injection technique (Stosiek et al., 2003). Astrocytes were counterstained with the red fluorescent marker sulforhodamine 101 (SR101) (Nimmerjahn et al., 2004). In this type of application, action potential-evoked calcium transients are measured as proxy for spiking activity in the local neuronal population (for review see Göbel and Helmchen, 2007a; Grewe and Helmchen, 2009; Lütcke and Helmchen, 2011).

In some of the galvanometric scanning modes, *xy*-scanners were combined with a piezo-electric objective focusing device for additional *z*-scanning. In all presented cases, scanner control signals were calculated on a PC and written to a commercially available digital-to-analog converter (DAC) card (Fig. 6A). Digitized PMT signals were integrated during the dwell time for each pixel on an FPGA module (Fig. 6). FPGA modules from National Instruments can easily be programmed using LabVIEW and enable high-speed signal conditioning without putting computational load onto the setup PC. As a cheaper, yet less powerful option, HelioScan can read in analog PMT signals using regular ADC cards.

The different imaging modes based on galvanometric mirrors mainly differ in the Trajectory component used. All of the Trajectory components are voxel components, implying that the trajectory coordinates and corresponding laser intensity values are computed on the PC rather than on the FPGA. The SignalWriter subcomponents of the employed ScanHead component (Fig. 6B) control the DAC card to generate the control signals for scanners and the intensity modulator based on the values calculated by the Trajectory component. The Clock subcomponent accesses the FPGA module via the FPGAWrapper subcomponent to generate the pixel clock signal that drives the generation of analog control signals by the DAC card. This way data generation and acquisition are synchronized. The DAQ TLC harbors SignalReader subcomponents that can read sample streams from different sources. One SignalReader reads, via its FPGAWrapper subcomponent, the integrated PMT values from the FPGA module. An instance of another SignalReader class, if required, reads in the position feedback values from a data acquisition card. Bundled into individual channels, PMT and position feedback values are forwarded to the ImageAssembler, where they are assembled into rectangular images. Images are then sent from the ImageAssembler to the Display to be drawn on the screen and to the DataCollection TLC for later retrieval and storage to disk. After image acquisition and upon user input, the DataCollection can stream previously recorded images to the Display for re-display.

2.5.1. *Frame scan mode*

In conventional frame scan mode, the specimen is scanned by the laser focus line by line within a horizontal plane. Laser trajectory coordinates can be modified in real-time for zooming and for rotating around the vertical *z*-axis. Continuous frame scanning with galvanometric scanners, i.e., taking a movie of fluorescence images, is the standard acquisition mode for calcium imaging (Fig. 6C and D). If required, the imaging plane can be tilted on the fly around a horizontal axis using a fast *z*-focusing device. Such an 'arbitrary plane imaging' approach allows en face viewing of oblique or vertical structures in the specimen (Fig. 6F) (Göbel and Helmchen, 2007b). With a motorized *z*-stage or a focusing device, standard *z*-stacks are acquired by repeatedly moving the focal plane in constant steps along the *z*-axis between frames.

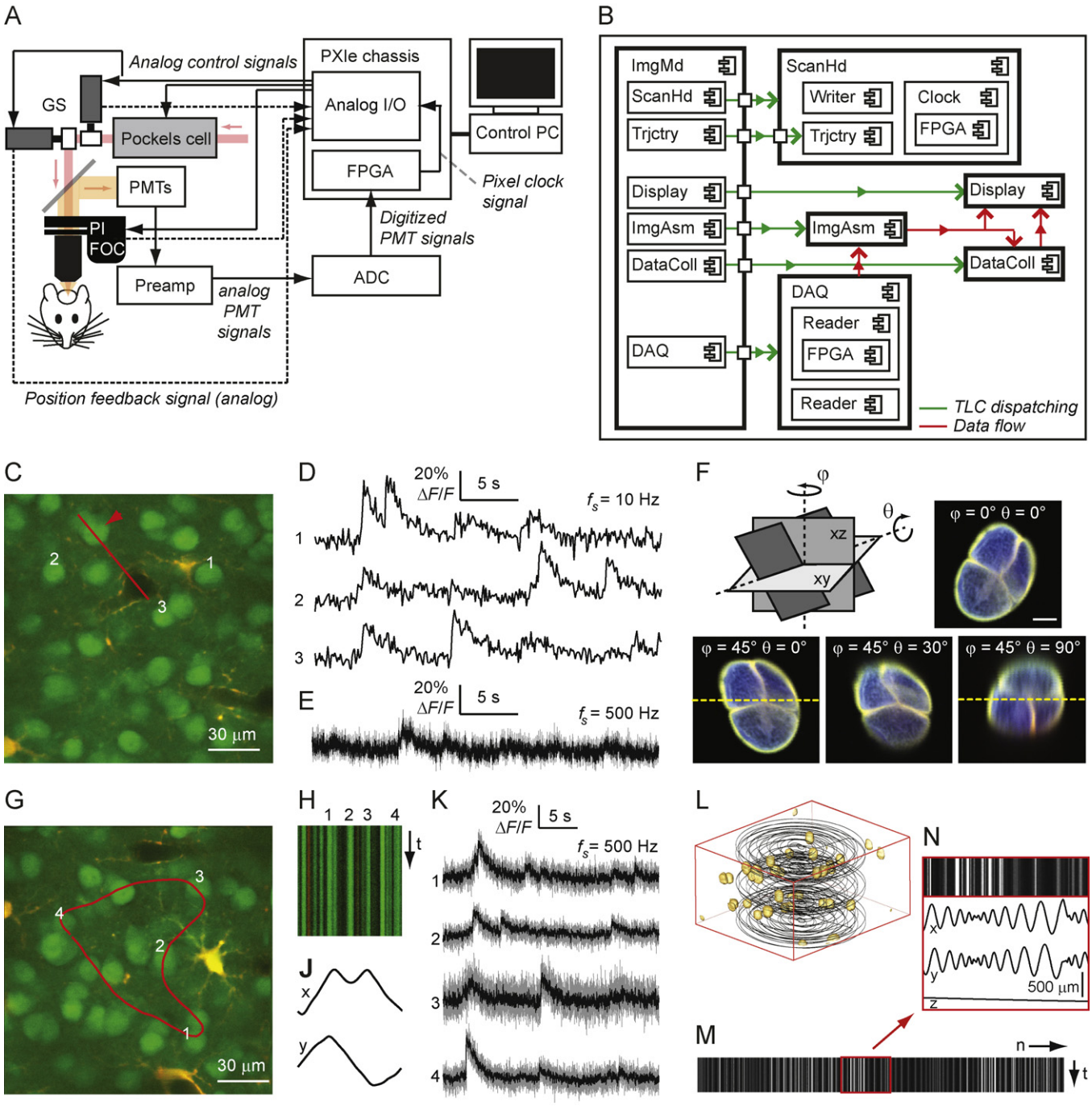

Fig. 6. Two-photon imaging modes based on galvanometric scan mirrors. (A) Hardware wiring diagram. Abbreviations: GS, galvanometric scan mirrors; PIFOC, piezo-electric focusing element; analog I/O, DAQ card; PMTs, photomultiplier tubes; ADC, analog-to-digital converter; FPGA, field-programmable gate array. (B) Interaction of the most important HelioScan components (TLCs with bold border, subcomponents with thin border). Component abbreviations: ImgMd, ImagingMode; Trjctry, Trajectory; ScanHd, ScanHead; FPGA, FPGAWrapper; Reader, SignalReader; ImgAsm, ImageAssembler. (C) Frame scan of a cell population in the barrel cortex of wild-type mouse, loaded with the calcium indicator OGB-1 (green) and SR101 (red). (D) Calcium transients from the cells marked in C, acquired using frame scanning (f_s, sampling frequency). (E) Calcium transients from the cell marked with red arrow in C, recorded using the straight line scan mode (scan trajectory shown in C). Unfiltered traces are shown in gray, filtered traces in black (box filter, width 50 samples). (F) Tilted frame scan of a pollen grain for different rotation angles. Scanning *z* direction is achieved by means of a PIFOC (see A). Scale bar 20 µm. (G) Reference image with overlaid user-defined arbitrary line scan visiting selected neurons. (H) PMT signals recorded during the arbitrary line scan depicted in G. (J) Scanner position signals recorded during the arbitrary line scan depicted in G. (K) Calcium transients from the cells marked in F, recorded in arbitrary line scan mode. Unfiltered traces are shown in gray, filtered traces in black (box filter of width 10). (L) Three-dimensional spiral scan allows sampling objects distributed in a volume. (M) PMT signals recorded during a 3D spiral scan of fluorescent beads in agarose gel. (N) Close-up of region marked by a red box in M, including corresponding scanner position signals. *n* refers to the pixel number and *t* to time. Inset shows the *x*–*y* and *z*-signals for a portion of the scan.

2.5.2. 2D line scan modes

With a two-dimensional frame-scan as reference image, features of interest can be selected and scanned at higher sampling rates using repetitive line scans. We implemented a simple line-scan mode, in which an arbitrary straight line is drawn by the user inside the reference frame. Scan speeds of 1 kHz and more are supported, for instance to acquire fast calcium transients from selected neurons (Fig. 6E). In a more sophisticated mode, several points of interest can be selected on the reference image and a smooth line is calculated for fast scanning (Fig. 6G) (Göbel and Helmchen, 2007b;

Göbel et al., 2007). Such user-defined line scans allow calcium measurements from preselected neuronal subsets at high temporal resolution (Fig. 6H–K). For all line scan modes, we acquired the position feedback signal from the scanners both for the reference image and the actual line scan (Fig. 6J) to perform post hoc pixel position assignment.

2.5.3. 3D spiral scan mode

In the living animal, cells are arranged in three-dimensional tissues. For neural tissue, in particular, it is desirable to observe activation patterns in neuronal populations distributed in 3D at sub-second temporal resolution. Acquiring conventional image stacks usually requires minutes per stack and is therefore too slow. By combining a sparse *x*–*y* sampling pattern, such as a spiral, with regular motion of a fast *z*-scanning device, a volume of a few hundred microns side-length can be sampled at several Hertz with high spatial coverage (Fig. 6L). HelioScan supports a 3D spiral mode where a spiral scan trajectory can be chosen based on a 3D reference image stack of the sample volume. Assignment of spiral pixels to voxels of the reference image stack is done based on the scanner position feedback signals, similar to the 2D line scan modes. 3D spiral scanning has been applied for example for in vivo calcium measurements of 3D neuronal network activity (Göbel et al., 2007).

2.6. Use case 3: In vivo two-photon imaging with acousto-optic deflectors

Various smart scan trajectories have been implemented with standard galvanometric scanners (Göbel and Helmchen, 2007b; Göbel et al., 2007; Lillis et al., 2008) with the aim to enable sampling of larger neuronal populations while retaining moderate sampling rates and sufficient dwell time per cell. Typically, however, the fundamental limitations in terms of maximum velocity and acceleration pertain. As an alternative, acousto-optic deflectors (AODs) lack the mechanical inertia and allow very fast (microseconds) laser beam deflection (Fig. 7A). Briefly, sound waves are generated by a piezo-transducer, coupled to and transmitted through a crystal, spatially modulating the refraction index of the crystal in a frequency-dependent manner and thus creating an optical diffraction grating. HelioScan supports two imaging modes for AOD-based laser-scanning microscopy, which differ in the ImagingMode and the Trajectory component used (Fig. 7B). In contrast to the galvanometric scanning modes, the AOD Trajectory components are FPGA trajectories. This implies that an FPGA module calculates *on the fly* the digital control signals for the digital synthesizer board (DDS) that generates the high-frequency signals for the AOD transducers (Fig. 7A). On the acquisition side, PMTs signals are processed similarly as in the modes presented in use case 2 using the same software components (Fig. 7B).

2.6.1. Frame scan mode

In this mode, rectangular field-of-views are scanned similar to the corresponding imaging mode for galvanometric scan mirrors (Fig. 7C). The properties of AODs bring about some differences from the user's perspective. The maximum field-of-view is typically smaller compared to galvanometric scanning due to the smaller maximum deflection angle of AODs. Furthermore, changes in the sound wave need to propagate through the entire aperture width of an AOD crystal before the full beam intensity arrives at the newly

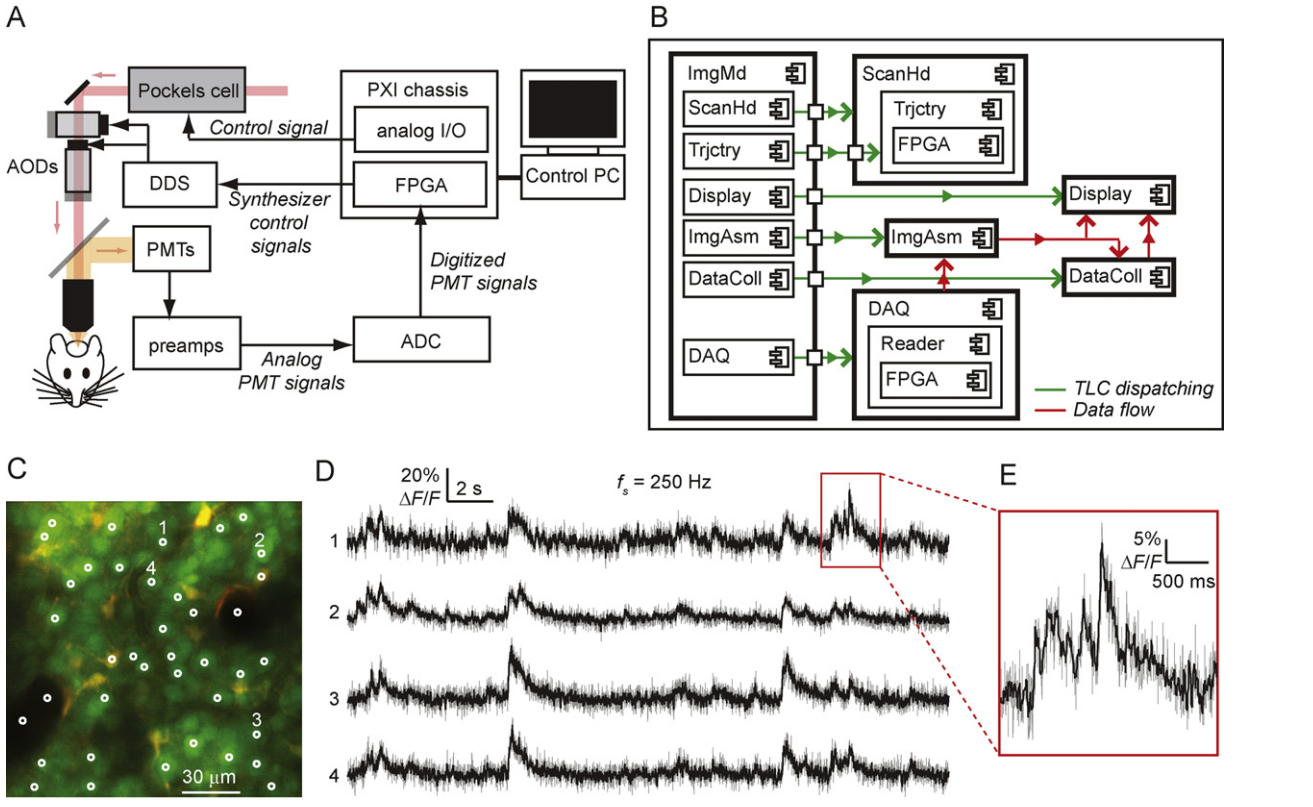

Fig. 7. Two-photon imaging modes based on AOD scanning. (A) Hardware wiring diagram. Abbreviations: DDS, digital synthesizer board. (B) Interaction of the most important HelioScan components (TLCs with bold border, subcomponents with thin border). Abbreviations: ImgMd, ImagingMode component; Trjctry, Trajectory; ScanHd, ScanHead; FPGA, FPGAWrapper; Reader, SignalReader; ImgAsm, ImageAssembler. (C) Reference image acquired in AOD frame scan mode from barrel cortex of a P14 wild-type mouse. All cells are unselectively loaded with the calcium indicator OGB-1 (green). Astrocytes are stained with SR101 (red). (D) Selection of calcium transients recorded in RAPS mode at 250 Hz sampling rate from the 40 locations marked in C. Unfiltered traces are shown in gray, filtered traces in black (box filter of width 5). Note the presence of synchronous bursts that is typical for cortical activity in mice of this age (Golshani et al., 2009; Rochefort et al., 2009).

addressed pixel. The user can exclude this 'equilibration time' from signal acquisition to improve spatial resolution, albeit then accepting a correspondingly lower signal-to-noise ratio for a given frame rate and *xy*-resolution.

2.6.2. Random-access scanning

The key advantage of AOD scanners is that the laser focus can be moved from one point of interest to the next in a random access manner (Fig. 7C). Depending on the pixel dwell time and the number of points to probe, such random access scanning enables measurements from user-selected sets of points at sampling rates of several hundred Hertz or even kilohertz, thus supporting high-speed calcium imaging (Fig. 7D) (Grewe et al., 2011). HelioScan also supports the so-called 'random access pattern scanning' (RAPS) mode (Grewe et al., 2011), in which a stereotype circular or spiral-shaped sub-pattern is defined (e.g., to cover a neuronal cell body) that is run on each predefined position (avoiding too long 'parking' of the laser spot, which may cause photo-bleaching or damage). In RAPS mode, signal integration is not halted during the equilibration time within a pattern but only when moving from one sub-pattern to the next. For each sub-pattern, the acquired signals can be averaged off-line and assigned to the respective cell.

2.7. Use case 4: In vivo two-photon imaging with a resonant scanner

Another method to achieve high imaging speed is the use of a resonant scanning system, which has recently gained interest in the two-photon imaging community (Bonin et al., 2011; Rochefort et al., 2009). While galvanometric scanners follow the externally

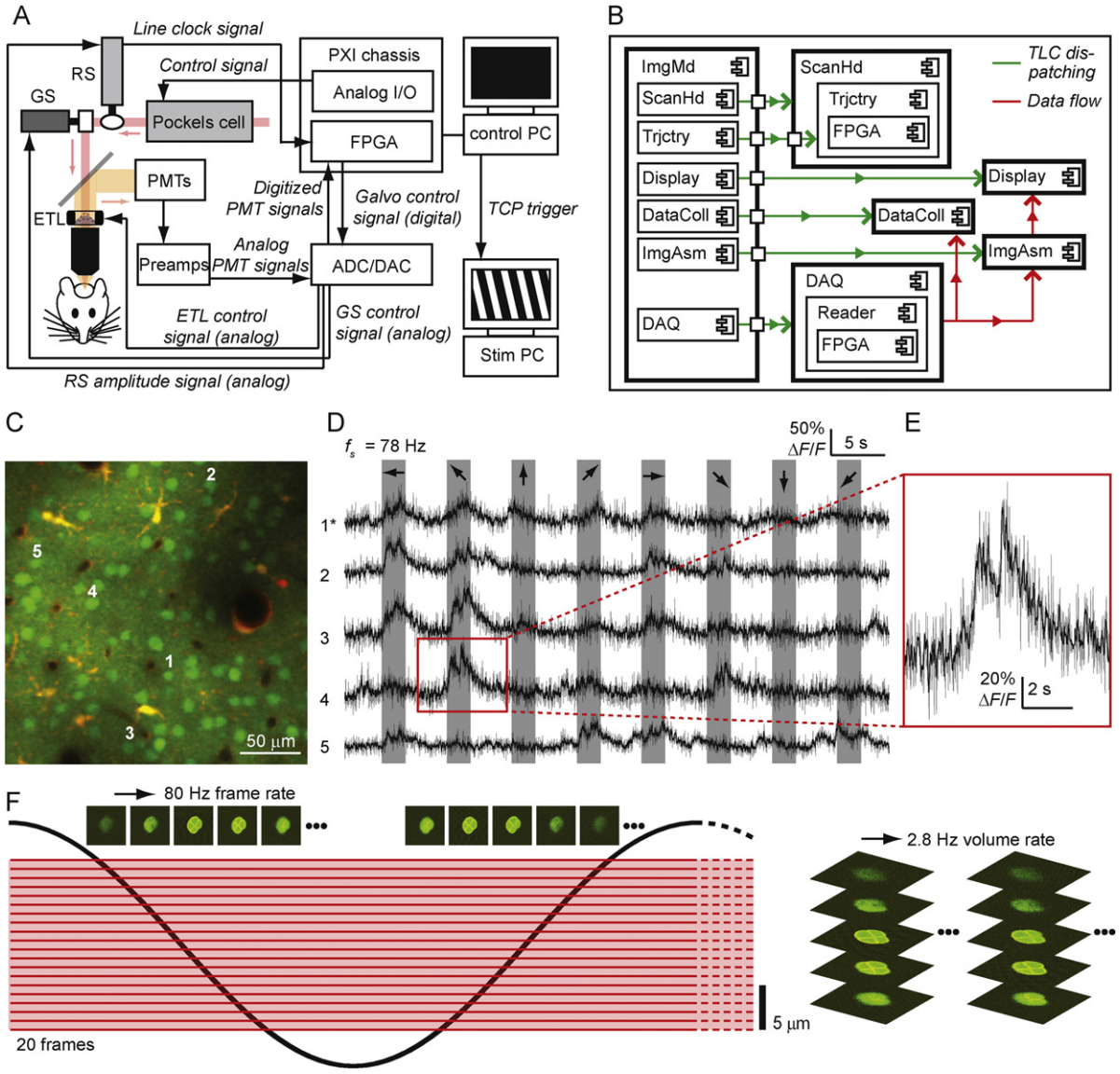

Fig. 8. Two-photon imaging modes based on a resonant scanner. (A) Hardware wiring diagram. Abbreviations: GS, galvanometric *y*-scan mirror; RS, resonant *x*-scanner; ADC/DAC, combined A/D-D/A converter; stim PC, PC generating visual stimuli. (B) Interaction of the most important HelioScan components (TLCs with bold border, subcomponents with thin border). Abbreviations: ImgMd, ImagingMode; Trjctry, Trajectory; ScanHd, ScanHead; FPGA, FPGAWrapper; Reader, SignalReader; ImgAsm, ImageAssembler; DataColl, DataCollection. (C) Fluorescence image (average of 1000 frames acquired at 78 Hz) of OGB-1 loaded neurons (green), and SR101-stained astrocytes (red) in layer 2/3 of visual cortex of an adult GAD67-GFP mouse. (D) Example calcium transients from cells marked in C, extracted from a frame series acquired at 78 Hz frame rate. Unfiltered traces are shown in gray, filtered traces in black (box filter of width 5). Gray areas indicate visual stimulation with gratings of different orientation (indicated by arrows). Cell 1 (trace marked with an asterisk) was GABAergic (i.e., expressing GFP), while cells 2–5 were non-GABAergic. (E) Expanded view of the calcium transient marked with a red box in D, presumably reflecting a burst of action potentials. (F) Volume scan of a fluorescently stained pollen grain, acquired with an ETL as fast *z*-focusing device (frame scan rate 78 Hz, volume scan rate 2.8 Hz, 20 *z*-planes with 1-µm step size).

applied control signal as close as possible given their frequency-response limitation, resonant scanners continually change the mirror orientation in a harmonic oscillation at their resonant frequency. This fixed motion pattern makes it necessary to trigger data acquisition by the line clock signal of the resonant scanner. Scanners with different resonance frequencies in the range of 4–16 kHz are available. The angular velocity at the quasi-linear part of their sinusoidal scan pattern typically is several-fold higher than the maximum achievable velocity with galvanometric scanners. The resulting high scan speed allows very fast line scans or frame scanning at video or higher rate (Leybaert et al., 2005; Nguyen et al., 2001).

In HelioScan, we implemented two modes for resonant-scanning systems, both employing the same software components (Fig. 8). Due to the high pixel rates achieved, the ImageAssembler was not able to assemble each image frame. Therefore, we implemented a DataCollection registering directly as an observer at the DAQ component and writing the raw pixel streams to disk. With this approach, the ImageAssembler can discard data when overloaded; still achieving video-rate at the Display component and without data loss in the raw data files.

2.7.1. Fast frame scan mode

We combined an 8-kHz resonant scanner for fast scanning in *x*-direction with a galvanometric scanner for slower scanning in *y*-direction. An FPGA was programmed to generate the control signal for the galvanometric scanner and digitize and integrate PMT signals, both time-locked to the line scan signal from the resonant scanner (Fig. 8A). The effective frame rate depended on the chosen number of lines per frame. For our example calcium imaging data, 200 lines were acquired at 78 Hz frame rate (Fig. 8C and D). Because of the fixed motion pattern of the resonant scanner, the laser focus trajectory cannot be rotated as with a galvanometric or AOD *xy*-scanner. Zooming-in is possible by adjusting the oscillation amplitude using an externally applied analog voltage controlled by HelioScan.

2.7.2. Volume scan mode

Resonant scanning also allows volume imaging at relatively high speed, which is desirable for example to measure population activity in neuronal networks. Employing a recently characterized electrically tunable lens (ETL) (Grewe et al., 2011) we extended the fast frame scan mode to 3D by dynamically adjusting the focal plane while scanning frames in *x* and *y* at high speed (Fig. 8F). The FPGA generated the control signals for both the ETL and the Pockels cell, dynamically adjusting the laser intensity according to *z*-depth. As in the previous frame-scan mode, the line scan signal of the resonant scanner served as the common timing source.

3. Discussion

We introduced HelioScan as a software framework that facilitates implementing and assembling control software for custom-built microscope systems. HelioScan copes with hardware and functional diversity by assembling at run-time from individually configurable software components. The high combinatorial flexibility is complemented by easy extendibility. Because of the highly structured approach and well-documented framework classes, new software components can be implemented rather quickly in understandable new source code. Moreover, the high modularity allows multiple developers to work in parallel on extending HelioScan functionality. HelioScan thus meets several of the challenges faced by microscope developers and users in general (Chi, 2008). Compared to other available open source microscopy software for laser-scanning microscopy (e.g., ScanImage, Pologruto et al., 2003; MPScope, Nguyen et al., 2009, 2006; or others, Hartell, 2007) or for IOS imaging (Harrison et al., 2009), HelioScan is not an individual software solution but rather provides a framework that supports many possible imaging systems. Another flexible software package for control of automated microscopes is μManager (Edelstein et al., 2010; Stuurman et al., 2007), which however does not support laser scanning systems. HelioScan thus fills a niche in the rapidly expanding area of research using specialized, custom-tailored intravital imaging systems.

In our four use cases, we demonstrated the potential of HelioScan by implementing various imaging modes with distinct requirements. The exemplified modes equip the user with a broad and state-of-the-art toolset for in vivo two-photon calcium imaging. Functional imaging studies of neuronal microcircuits in the brain are rapidly expanding, aiming at revealing principles of neural coding. Depending on the specific goals (e.g., large population measurements vs. high-speed recordings) different scanning modes are required, such as complex 2D or 3D trajectories (Göbel and Helmchen, 2007b; Göbel et al., 2007; Katona et al., 2011; Lillis et al., 2008), resonant scanning (Nguyen et al., 2001; Bonin et al., 2011; Keller et al., 2012; Rochefort et al., 2009), or the use of acousto-optic scanners in one spatial dimension (Chen et al., 2011, 2012), in 2D (Grewe et al., 2010; Otsu et al., 2008; Salome et al., 2006), or even in 3D (Duemani Reddy et al., 2008; Katona et al., 2012; Kirkby et al., 2010). Moreover, multiplexing technology has been employed to enable fast in vivo two-photon imaging from multiple field-of-views (Cheng et al., 2011) and simple tunable lenses may be used in combination with various *x*–*y* scanners to make 3D measurements (Grewe et al., 2011). Our aim is that all these variations of in vivo calcium imaging systems, and further ones likely to come up in the near future, can be supported by HelioScan. Likewise, it should be possible to integrate other functionality into HelioScan, for example the acquisition of electrophysiological data or the control of photostimulation devices, for which specialized software has been written (Bendels et al., 2008; Suter et al., 2010).

3.1. Current limitations

For any software framework, the developer has to comply with the structure enforced by the framework. Although the HelioScan framework architecture has coped with all requirements encountered so far, as demonstrated in our four use cases, we have also identified limitations of the current HelioScan framework architecture, which we plan to overcome in next major releases. In the following, we briefly discuss these limitations together with their implications and possible work-arounds for the current version of HelioScan.

Since flexibility has been the most important design goal, HelioScan is rather complex, which can be intimidating to programmers used to software that is not object-oriented. New developers can familiarize themselves with the HelioScan principles either in a *top-down* or *bottom-up* approach. We recommend newcomers to start bottom-up by implementing a simple new subcomponent, for which no knowledge about interacting TLCs is required. This way, developers can familiarize themselves with the basic features of a HelioScan component step by step. In a next phase, new developers can start to understand a HelioScan application in a top-down approach. They need to familiarize himself on an abstract level with the purpose of the individual component types and the way they interact. Subsequently, they can start to dive down into the specific class hierarchy of the component type they are interested in. Thus, despite the apparent complexity, once developers have familiarized themselves with the HelioScan framework structure, gained own experience and understood the limited set of rules, the framework structure will work in their favor.

The core framework structure of HelioScan is given by a limited number of defined TLC component types (Fig. 2B) and their modes

of interaction. Additional TLC component types would require a significant modification at the framework level, which would need to be well justified and orchestrated.

Only one instance per type of TLC is allowed at a time (Fig. 2B). For example, it is currently not possible to run multiple Stimulator instances simultaneously to apply different stimuli at the same time. Rather, one would currently have to implement a Stimulator component that is able to simultaneously deliver multiple stimuli at the same time, or, alternatively, trigger external stimulator devices using a TTL signal generated by HelioScan (e.g., the shutter signal).

Data streaming connections (e.g., from DAQ to ImageAssembler; Figs. 5B, 6B, 7B, and 8B) are currently hardwired on the side of data-receiving TLCs (i.e., observers). As a consequence, the topology of data streaming results from the used TLCs and cannot be changed at configuration time. However, since this is not in principle enforced by the framework, developers are free to make it a configurable feature of their own components.

Coupling refers to the extent to which a component relies on other components for its function. By following a component-oriented approach, we tried to keep coupling as low as possible. Interaction of components, be it by method calls or exchange of triggers, involves the interfaces defined at the generic level of the involved component types (Fig. 2A) (with an exception of the previously mentioned adapter classes). The coupling of TLCs due to call of method VIs in some cases limits their testability because the functionality of a TLC may at least partly depend on the call of methods provided by other TLCs. In the current implementation of HelioScan, the call of other methods belonging to other TLCs are conditionally disabled in stand-alone test VIs; only in the HelioScan main VI, the required conditional enable symbols are defined such that the enclosed methods calls are executed.

A HelioScan application requires all involved components to load locally on a single computer. In cases where peripheral hardware or computational resources need to be accessed via additional computers, it would be desirable to overcome this limitation. National Instruments provides different technologies that can be used for this purpose. Communication to LabVIEW software running on a remote computer can be achieved by means of so-called shared variables (for synchronous access) or network streaming queues (for asynchronous, buffered access). Simple and parallelizable computation involving fast hardware access can be executed on FPGA modules from National Instruments (Figs. 6A, 7A, and 8A). The corresponding bit files can be easily created and accessed from within LabVIEW and deployed by HelioScan components (typically FPGAWrapper components, Figs. 6B, 7B, and 8B) on the fly. National Instruments real-time (RIO) modules are hardware systems running a deterministic real-time operating system and can be programmed with LabVIEW, as well as accessed from within HelioScan.

Currently, the HelioScan framework and therefore all available components are exclusively written in LabVIEW. LabVIEW supports the call of functions written in other languages via DLLs, .NET or ActiveX. Also MATLAB code can be executed. An alternative approach is to communicate with other programs via TCP/IP (Fig. 7A). Another important aspect concerns the fact that the LabVIEW development environment is commercial software and is not available for free. For laboratories that do not own a LabVIEW license and are not interested in pursuing development work, there is the option to use a HelioScan executable, which only requires run-time engines available at a significantly lower cost.

3.2. Outlook

Starting from the limitations of the current architecture of HelioScan as discussed above, we concluded the following list of requirements for the next generation of our software framework. First, developers should not be restricted anymore to pre-defined TLC component types. Second, it should be possible to run multiple instances of the same TLC simultaneously. Third, arbitrary data streaming networks should be configurable, rather than data streaming directions being hardwired as it is currently the case (compare Figs. 5B, 6B, 7B, and 8B). Fourth, coupling between components should be further decreased to allow for better unit tests. Fifth, it should be possible to run the TLCs of a HelioScan application in a distributed fashion on several computers. Finally, implementation of TLCs should not be limited to LabVIEW. As a future outlook, we are currently planning a new architecture for HelioScan to fit these new requirements. In the envisioned approach, TLCs will communicate by exchanging XML-based messages via a message-oriented middleware. On each computer enabled for HelioScan, a component manager will be running. Driven by a configuration file, the component manager will boot up the specified TLC instances, independent of the language they have been written in. The boot-up process for each component will occur either locally or remotely (via the component managers running on other computers), just as specified in the configuration file. Using such a middleware-oriented architecture, components will be highly decoupled. It will thus be much better possible to develop unit tests for components, since their whole environment can be simulated by appropriate XML-based messages.

In conclusion, due to its unprecedented flexibility, we expect HelioScan to become a highly valuable microscopy software package for in vivo imaging laboratories that have to deal with diverse or frequently changing hardware and functionality requirements. With HelioScan tailored to be easily extendible, we expect the user community to contribute their own components and extend the available toolsets accordingly. HelioScan can then not only keep track with the technological advancements in the research fields, but also serve as a software environment suitable to realize innovative imaging methods. HelioScan can be obtained for free. Information on how to get started can be found on the HelioScan website (www.helioscan.org).

4. Materials and methods

4.1. Microscope control PC

A PC (Dalco AG, Wilen, Switzerland) with the following features was used: an Intel Core i7 960 CPU (3.20 GHz), an ASRock X58 Deluxe3 mainboard, 6 GB DDR3-SDRAM (1333 MHz) and a NVIDIA Quadro NVS 295 graphics card. The Windows 7 Enterprise 64 bit operating system (Microsoft, Redmond, WA) was installed with the following additional software: LabVIEW 2010 SP1, FPGA module, Xilinx FPGA Tools, Vision Development Module and device drivers (all from National Instruments, Ennetbaden, Switzerland), VI Package Manager Community Edition with easyXML (both from JKI, Walnut Creek, CA), Git and TortoiseGit.

4.2. Intrinsic optical signal (IOS) imaging

We integrated a system for IOS imaging into a two-photon microscope setup. Image acquisition was performed with a 12 bit gray scale CCD camera (Teli CS3960DCL; Thoshiba Teli Corporation, Tokyo, Japan) interfaced to the setup PC by a camera link card (NI PCI-1426, National Instruments). We used a 4× objective (UPlanFLN, 4×/0.13; Olympus, Tokyo, Japan) to which a ring harboring LEDs for illumination was attached. HelioScan in video camera mode was used to acquire a blood vessel reference map under green light illumination (peak wavelength 525 nm, L5-G61N-GT LED; Sloan LED, Delft, The Netherlands). The component classes

Table 1
HelioScan component classes used for video camera mode (VC) and intrinsic optical signal (IOS) imaging mode, respectively. Subcomponents are indented and listed under the class they are owned by. Only relevant TCLs are listed.

Component class	Mode
GenericExperimentController	VC
ExperimentControllerDL090130Interval	IOS
GenericSweep	IOS
ImagingModeMG090624Camera	VC
ImagingModeMG091111IntrinsicImaging	IOS
GenericScanHead	VC/IOS
DAQ_MG090622Camera	VC/IOS
ImageAssemblerMG090623Binning	VC/IOS
DataCollectionDL090215	VC
DataCollectionMG091001IntrinsicImaging	IOS
DisplayDL090216	VC/IOS
ImageProcessorDL090216RangeOffset	VC/IOS
GenericStimulator	VC
StimulatorDL090503	IOS
SignalWriterDL090202DAQmx	IOS

HelioScan was configured to run with in this mode are listed in Table 1 (compare Fig. 5B).

Subsequent IOS imaging was performed under red light illumination (peak wavelength 660 nm, L-7113SRD LED; Kingbright Electronic, Taipai, Taiwan). See Table 1 for the component classes HelioScan was configured for in IOS mode (compare Fig. 5B). Whiskers were stimulated with rostro-caudal motion using a piezo-bender actuator (PL140.10, PI Ceramic GmbH, Lederhose, Germany). Drive signals for sinusoidal stimulation (10 Hz) were generated with a DAC card (NI PXI-6229, National Instruments) and amplified appropriately (using a Dual Piezo Amplifier from Sigmann Elektronik GmbH, Hüffenhardt, Germany). An individual measurement (sweep) consisted of three image acquisition phases A, B and C, each with the same duration. Only phase C was acquired with simultaneous sensory stimulation of the animal. For each time-point relative to the start of a phase, an image with the relative difference between B and A, and C and B, respectively, was calculated. For space-resolved IOS imaging, difference images were averaged over the duration of a phase. Both for space- and time-resolved IOS imaging, sweeps were repeated and their results averaged until a clear intrinsic signal was obtained.

4.3. Two-photon microscopy using galvanometric mirrors

A custom-built two-photon microscope powered by a Ti:Sapphire laser (MaiTai Broadband, Spectra-Physics, Santa Clara, CA) was used. Beam-size was adjusted with a telescope and laser intensity modulated with a Pockels cell (model 350/80, with controller model 302RM; Conoptics, Danbury, CT). Two galvanometric mirrors (model 6210; Cambridge Technology, Lexington, MA) were used for *x*/*y* scanning and a piezoelectric focusing device (P-725.4CD PIFOC; Physik Instrumente, Karlsruhe, Germany) for stack acquisition and 3D scanning. Fluorescence signals were detected with PMTs (R6357; Hamamatsu Photonics, Hamamatsu City, Japan), the gain of which was controlled by custom-built power supplies (based on C4900-01, Hamamatsu). PMT signals were pre-amplified with transimpedance amplifiers (150 kΩ, 1 MHz; XPG-ADC-PREAMP from Sigmann Elektronik GmbH, Hüffenhardt, Germany) and fed into a multichannel ADC (details see below).

Two different hardware architectures for signal acquisition were realized. In the first architecture ('CSEM system'), the setup PC was interfaced to a PXI chassis (NI PXI-1036) from National Instruments harboring the following modules: an analog I/O card (NI PXI-6259) controlling galvanometric *x*/*y* scan mirrors, the *z*-focusing device and the Pockels cell; and an NI PXI-7813R FPGA module for real-time integration of digitized PMT signals. NI BNC-2110 connector boxes were used to connect periphery devices to the analog I/O card. For experiments involving the 2D arbitrary line scan and 3D spiral scan mode, the NI PXI-6259 card was also used for acquisition of analog position-feedback signals from the galvanometric scanners and the *z*-focusing device. A custom-built ADC (CSEM, Neuchâtel, Switzerland) with four analog input channels (based on the LTC2247 ADC chip; 14 bit, 40 MS/s; from Linear Technology, Milpitas, CA) digitized the PMT signals and fed them into the FPGA module for intra-pixel integration.

Table 2
HelioScan component classes used for different galvanometric scanning modes: frame scan mode (FS), straight line scan mode (SLS), arbitrary line scan mode (ALS) and three-dimensional spiral scan mode (SS). Subcomponents are indented and listed under the class they are owned by. Note the use of different FPGAWrapper classes for the CSEM system[1] and the FlexRIO system[2]. Only relevant TCLs are listed.

Component class	Mode
ImagingModeDL090130Frame	FS
TrajectoryDL090201FrameScan	FS
ImagingModeDL100901FlexTrajectory	SLS/ALS/SS
TrajectoryDL110413StraightLine	SLS
TrajectoryDL120126ArbitraryLine2D	ALS
TrajectoryCN090716SpiralScan	SS
ScanHeadDL090202	FS/SLS/ALS/SS
SignalWriterDL090202DAQmx	FS/SLS/ALS/SS
GenericDigitalOut	FS/SLS/ALS/SS
ClockDL090202FPGA	FS/SLS/ALS/SS
FPGAWrapperDL090211Galvo	FS[1]/SLS[1]/ALS[1]/SS
FPGAWrapperAN110607Galvo	FS[2]/SLS[2]/ALS[2]/SS
DAQ_DL091205Readers	FS/SLS/ALS/SS
SignalReaderDL091205FPGA	FS/SLS/ALS/SS
FPGAWrapperDL090211Galvo	FS[1]/SLS[1]/ALS[1]/SS
FPGAWrapperAN110607Galvo	FS[2]/SLS[2]/ALS[2]/SS
SignalReaderCN091129DAQmx	SLS/ALS/SS
ImageAssemblerDL090211Frame	FS/SLS/ALS/SS
DataCollectionDL090215	FS/SLS/ALS/SS
DisplayDL090216	FS/SLS/ALS/SS
ImageProcessorDL090216RangeOffset	FS/SLS/ALS/SS

In the second architecture ('FlexRIO system'), the setup PC interfaced to a PXIe chassis (NI PXIe-1079) harboring an NI PXI-6259 card for scan signal generation and acquisition of position-feedback, and a NI PXIe-7962R FlexRIO FPGA module connected to an NI 5751 FlexRIO adapter module (16 analog input channels, 14 bit resolution, sampled at 50 MS/s) for PMT signal acquisition and intra-pixel integration. An NI SMB-2147 and an NI SMB-2148 accessory were used to connect the FlexRIO adapter module to pre-amplified PMT signal inputs. The HelioScan component classes used for the different imaging modes are listed in Table 2 (compare Fig. 6B).

4.4. Two-photon microscopy using acousto-optical deflectors

The AOD-based microscope (for details see Grewe et al. (2010)) differed in the following aspects from the one described in the previous paragraph. The laser beam of a Ti:Sapphire laser (Chameleon Ultra II, Coherent, Santa Clara, CA) was expanded with a variable beam expander (S6ASS2075, Silloptics, Wendelstein, Germany). Beam deflection was achieved with a pair of orthogonally mounted AODs (DTSXY-A12-850, A&A Optoelectronics, Markham, Ontario, Canada). Spatial dispersion caused by the AOD crystals was pre-compensated using a prism (N-SF6, apex angle 60°, uncoated; Thorlabs) (Grewe et al., 2011). Pre-amplified PMT signals were digitized and integrated using the above-mentioned custom-built ADC

Table 3
HelioScan component classes used for the two AOD-based scan modes: frame scan mode (FS) and RAPS mode. Subcomponents are indented and listed under the class they are owned by. Only relevant TCLs are listed.

Component class	Mode
ImagingModeDL090130Frame	FS
TrajectoryMG100122AODFrameScan	FS
FPGAWrapperMH120124FrameScan	FS
ImagingModeDL100901FlexTrajectory	RAPS
TrajectoryMH100122AODRAPS	RAPS
FPGAWrapperMH120124RAPS	RAPS
ScanHeadMH100729FPGA	FS/RAPS
AnalogOutDL101207DAQmx	FS/RAPS
DAQ_DL091205Readers	FS/RAPS
SignalReaderDL091205FPGA	FS/RAPS
FPGAWrapperMH120124FrameScan	FS
FPGAWrapperMH120124RAPS	RAPS
ImageAssemblerDL090211Frame	FS/RAPS
DataCollectionDL090215	FS/RAPS
DisplayDL090216	FS/RAPS
ImageProcessorDL090216RangeOffset	FS/RAPS

interface to an NI PXI-7813R FPGA module in a NI PXI-1036 chassis. The FPGA module was also used for real-time control of a synthesizer board (AD9852, Analog Devices GmbH, Munich, Germany) generating the oscillations for the AOD transducers. The Pockels cell control voltage was generated using a NI PXI-6229 card harbored in the PXI chassis. The HelioScan component classes used for scanning with AODs are listed in Table 3 (compare Fig. 7B).

4.5. Two-photon microscopy using a resonant scanner

We used a microscope similar to the one with galvanometric scanners described above. A resonant scanner (CRS Series, 8 kHz; Cambridge Technology) was used for fast scanning in *x* direction. Scanning in *y* direction was achieved with a galvanometric scan mirror (model 6210; Cambridge Technology). PMT signals were pre-amplified with a variable gain high speed current amplifier (DHPCA-100; Femto, Berlin, Germany), digitized with a modified version of the custom-built ADC (CSEM, Neuchâtel, Switzerland) and fed into a NI PXI-7813R FPGA module in a NI PXI-1036 chassis. In addition to the four analog input channels, this ADC version featured seven analog output channels (12 bit resolution; based on the AD5328 chip from Analog Devices GmbH, Munich, Germany) controllable by the FPGA module. The FPGA performed intra-pixel integration of the PMT signals during the quasi-linear fraction of both fly-forward and fly-back of the sinusoidal motion of the resonant scanner. Using one of the seven analog output channels, the FPGA generated a step-wise scan pattern controlling the galvanometric *y* scan mirror. The line-clock signal provided by the control board of the resonant scanner (model 311-14988-7, Cambridge Technology) was used to synchronize the scan and acquisition process. The analog voltages controlling the oscillation amplitude of the resonant scanner and the Pockels cell for laser intensity control, respectively, were generated using an NI PXI-6229 card. In volume scan mode, we used an electrically tunable lens (EL-C-10-30-VIS-LD, Optotune, Dietikon, Switzerland) combined with a plano-concave offset lens with −100 mm focal length (Thorlabs) (Grewe et al., 2011). We used HelioScan configured to run with the components listed in Table 4 (compare Fig. 8B).

4.6. Mouse preparation and calcium indicator labeling

All animal procedures were carried out according to the guidelines of the Center for Laboratory Animals of the University of Zurich and were approved by the Cantonal Veterinary Office. C57BL/6 wild-type mice (use case 1, 2 months old; use cases 2 and 3, 14 days old) and GAD67-GFP mice (use case 4, 2 months old) (Tamamaki et al., 2003) were anesthetized with isoflurane (1–2% in oxygen) and a craniotomy was prepared above the somatosensory cortex as previously described (Langer and Helmchen, 2012). The dura was carefully removed and the exposed cortex was superfused with normal rat Ringer (NRR) solution (135 mM NaCl, 5.4 mM KCl, 5 mM Hepes, 1.8 mM $CaCl_2$; pH 7.2 with NaOH). Cell populations in cortical layer 2/3 were labeled with the calcium indicator Oregon Green BAPTA-1 AM (Molecular Probes, Invitrogen) (Stosiek et al., 2003). Briefly, 50 μg of OGB-1 AM were dissolved in DMSO plus 20% Pluronic F-127 (BASF) and diluted in NRR to a final concentration of about 1 mM. This solution was pressure-ejected into the tissue using a micropipette. Brief (5–10 min) application of SR101 (50 μM in NRR) to the exposed neocortical surface resulted in co-labeling of the astrocytic network (Nimmerjahn et al., 2004). To dampen heartbeat- and breathing-induced motion, the cranial window was filled with agarose (type III-A, Sigma; 1% in NRR) and covered with an immobilized glass cover slip.

Table 4
HelioScan component classes used for the frame scan mode (FS) and volume scan mode (VS) based on a resonant scanner. Only relevant TCLs are listed.

Component class	Mode
ImagingModeDL090130Frame	FS, VS
TrajectoryAK101022Resonance	FS, VS
FPGAWrapperAK101022Resonance	FS, VS
ScanHeadMH100729FPGA	FS, VS
AnalogOutDL101207DAQmx	FS, VS
DAQ_DL091205Readers	FS, VS
SignalReaderDL091205FPGA	FS, VS
FPGAWrapperAK101022Resonance	FS, VS
ImageAssemblerDL090211Frame	FS, VS
DataCollectionDL110128Streaming	FS, VS
DisplayDL090216	FS, VS
ImageProcessorDL090216RangeOffset	FS, VS

Author contributions

HK had the original idea of a plug-in-oriented software architecture. DL designed and implemented most of the HelioScan framework and coordinated collaboration efforts. MvH and DL implemented the AOD-based imaging modes. AJK and DL implemented the resonant-scanner-based frame scan imaging, OAP and DL the resonant-scanner-based volume scanning. DL implemented the galvanometric mirror-based 2D scan modes. CN and DL implemented the galvanometric-mirror-based 3D scanning. MG and DL implemented the camera-based imaging modes. HK planned and built various electronic components for the microscope setups. DL and FH wrote the manuscript and designed the figures.

Acknowledgements

We would like to thank Stefan Giger for constructing and building various parts of our microscope setups, Samuel Häusler for implementing a software interface for the digital-to-analog converter used for resonant scanner imaging, and Marco Tedaldi for his help with bug hunting. We thank Klas Kullander from Uppsala University for support of CN. We are also grateful to Roland Krüppel, Yves Kremer and Carl Petersen for their valuable feedback on the manuscript. This work was supported by a PhD fellowship to DL from the Roche Research Foundation and grants to FH from the Swiss National Science Foundation (SNSF; grants

310030-127091), the German-Swiss Research Group "Barrel Cortex Function" (FOR1341, SNSF 310030-130826), the EU-FP7 program (BRAIN-I-NETS project 243914), and the Swiss SystemsX.ch Initiative (project 2008/2011-Neurochoice).

References

Bendels MH, Beed P, Leibold C, Schmitz D, Johenning FW. A novel control software that improves the experimental workflow of scanning photostimulation experiments. J Neurosci Methods 2008;175:44–57.
Blume PA. The LabVIEW style book. Upper Saddle River, NJ: Prentice Hall; 2007.
Bonin V, Histed MH, Yurgenson S, Reid RC. Local diversity and fine-scale organization of receptive fields in mouse visual cortex. J Neurosci 2011;31:18506–21.
Chen X, Leischner U, Rochefort NL, Nelken I, Konnerth A. Functional mapping of single spines in cortical neurons in vivo. Nature 2011;475:501–5.
Chen X, Leischner U, Varga Z, Jia H, Deca D, Rochefort NL, et al. LOTOS-based two-photon calcium imaging of dendritic spines in vivo. Nat Protoc 2012;7:1818–29.
Cheng A, Goncalves JT, Golshani P, Arisaka K, Portera-Cailliau C. Simultaneous two-photon calcium imaging at different depths with spatiotemporal multiplexing. Nat Methods 2011;8:139–42.
Chi KR. Imaging and detection: focusing on software. Nat Methods 2008;5:651–3.
Denk W, Strickler JH, Webb WW. Two-photon laser scanning fluorescence microscopy. Science 1990;248:73–6.
Dombeck DA, Khabbaz AN, Collman F, Adelman TL, Tank DW. Imaging large-scale neural activity with cellular resolution in awake, mobile mice. Neuron 2007;56:43–57.
Duemani Reddy G, Kelleher K, Fink R, Saggau P. Three-dimensional random access multiphoton microscopy for functional imaging of neuronal activity. Nat Neurosci 2008;11:713–20.
Edelstein A, Amodaj N, Hoover K, Vale R, Stuurman N. Computer control of microscopes using micro-Manager. Curr Protoc Mol Biol 2010;2010:14.20.1–17.
Engelbrecht CJ, Johnston RS, Seibel EJ, Helmchen F. Ultra-compact fiber-optic two-photon microscope for functional fluorescence imaging in vivo. Opt Express 2008;16:5556–64.
Frostig RD, Lieke EE, Ts'o DY, Grinvald A. Cortical functional architecture and local coupling between neuronal activity and the microcirculation revealed by in vivo high-resolution optical imaging of intrinsic signals. Proc Natl Acad Sci USA 1990;87:6082–6.
Göbel W, Helmchen F. In vivo calcium imaging of neural network function. Physiology 2007a;22:358–65.
Göbel W, Helmchen F. New angles on neuronal dendrites in vivo. J Neurophysiol 2007b;98:3770–9.
Göbel W, Kampa BM, Helmchen F. Imaging cellular network dynamics in three dimensions using fast 3D laser scanning. Nat Methods 2007;4:73–9.
Golshani P, Goncalves JT, Khoshkhoo S, Mostany R, Smirnakis S, Portera-Cailliau C. Internally mediated developmental desynchronization of neocortical network activity. J Neurosci 2009;29:10890–9.
Grewe BF, Helmchen F. Optical probing of neuronal ensemble activity. Curr Opin Neurobiol 2009;19:520–9.
Grewe BF, Langer D, Kasper H, Kampa BM, Helmchen F. High-speed in vivo calcium imaging reveals neuronal network activity with near-millisecond precision. Nat Methods 2010;7:399–405.
Grewe BF, Voigt FF, van't Hoff M, Helmchen F. Fast two-layer two-photon imaging of neuronal cell populations using an electrically tunable lens. Biomed Opt Express 2011;2:2035–46.
Grinvald A, Sharon D, Slovin H, Vanzetta I. Intrinsic signal imaging in the neocortex: implications for hemodynamic-based function imaging. In: Yuste R, Konnerth A, editors. Imaging in neuroscience and development: a laboratory manual. Woodbury, NY: Cold Spring Harbor Laboratory Press; 2005.
Harrison TC, Sigler A, Murphy TH. Simple and cost-effective hardware and software for functional brain mapping using intrinsic optical signal imaging. J Neurosci Methods 2009;182:211–8.
Hartell NA. Simple Windows-based software for the control of laser scanning confocal microscopes. J Neurosci Methods 2007;162:26–31.
Harvey CD, Coen P, Tank DW. Choice-specific sequences in parietal cortex during a virtual-navigation decision task. Nature 2012;484:62–8.
Helmchen F, Denk W. Deep tissue two-photon microscopy. Nat Methods 2005;2:932–40.
Huber D, Gutnisky DA, Peron S, O'Connor DH, Wiegert JS, Tian L, et al. Multiple dynamic representations in the motor cortex during sensorimotor learning. Nature 2012;484:473–8.
Katona G, Kaszas A, Turi GF, Hajos N, Tamas G, Vizi ES, et al. Roller Coaster Scanning reveals spontaneous triggering of dendritic spikes in CA1 interneurons. Proc Natl Acad Sci USA 2011;108:2148–53.
Katona G, Szalay G, Maak P, Kaszas A, Veress M, Hillier D, et al. Fast two-photon in vivo imaging with three-dimensional random-access scanning in large tissue volumes. Nat Methods 2012;9:201–8.
Keller GB, Bonhoeffer T, Hubener M. Sensorimotor mismatch signals in primary visual cortex of the behaving mouse. Neuron 2012;74:809–15.
Kerr JN, Greenberg D, Helmchen F. Imaging input and output of neocortical networks in vivo. Proc Natl Acad Sci USA 2005;102:14063–8.
Kirkby PA, Srinivas Nadella KM, Silver RA. A compact Acousto-Optic Lens for 2D and 3D femtosecond based 2-photon microscopy. Opt Express 2010;18:13721–45.
Land PW, Simons DJ. Cytochrome oxidase staining in the rat SmI barrel cortex. J Comp Neurol 1985;238:225–35.
Langer D, Helmchen F. Post hoc immunostaining of GABAergic neuronal subtypes following in vivo two-photon calcium imaging in mouse neocortex. Pflügers Arch 2012;463:339–54.
Leybaert L, de Meyer A, Mabilde C, Sanderson MJ. A simple and practical method to acquire geometrically correct images with resonant scanning-based line scanning in a custom-built video-rate laser scanning microscope. J Microsc 2005;219:133–40.
Lillis KP, Eng A, White JA, Mertz J. Two-photon imaging of spatially extended neuronal network dynamics with high temporal resolution. J Neurosci Methods 2008;172:178–84.
Lütcke H, Helmchen F. Two-photon imaging and analysis of neural network dynamics. Rep Prog Phys 2011:74.
Margolis DJ, Lütcke H, Schulz K, Haiss F, Weber B, Kugler S, et al. Reorganization of cortical population activity imaged throughout long-term sensory deprivation. Nat Neurosci 2012;15:1539–46.
Nguyen Q-T, Driscoll J, Dolnick EM, Kleinfeld D. MPScope 2.0: a computer system for two-photon laser scanning microscopy with concurrent plasma-mediated ablation and electrophysiology. In: Frostig R, editor. In vivo optical imaging of brain function. Boca Raton, FL: CRC Press; 2009. p. 117–42.
Nguyen QT, Callamaras N, Hsieh C, Parker I. Construction of a two-photon microscope for video-rate Ca^{2+} imaging. Cell Calcium 2001;30:383–93.
Nguyen QT, Tsai PS, Kleinfeld D. MPScope: a versatile software suite for multiphoton microscopy. J Neurosci Methods 2006;156:351–9.
Nimmerjahn A, Kirchhoff F, Kerr JN, Helmchen F. Sulforhodamine 101 as a specific marker of astroglia in the neocortex in vivo. Nat Methods 2004;1:31–7.
Ohki K, Chung S, Ch'ng Y, Kara P, Reid R. Functional imaging with cellular resolution reveals precise micro-architecture in visual cortex. Nature 2005;433:597–603.
Otsu Y, Bormuth V, Wong J, Mathieu B, Dugue GP, Feltz A, et al. Optical monitoring of neuronal activity at high frame rate with a digital random-access multiphoton (RAMP) microscope. J Neurosci Methods 2008;173:259–70.
Pittet MJ, Weissleder R. Intravital imaging. Cell 2011;147:983–91.
Pologruto TA, Sabatini BL, Svoboda K. ScanImage: flexible software for operating laser scanning microscopes. Biomed Eng Online 2003;2:13.
Ranganathan GN, Koester HJ. Optical recording of neuronal spiking activity from unbiased populations of neurons with high spike detection efficiency and high temporal precision. J Neurophysiol 2010;104:1812–24.
Rochefort NL, Garaschuk O, Milos RI, Narushima M, Marandi N, Pichler B, et al. Sparsification of neuronal activity in the visual cortex at eye-opening. Proc Natl Acad Sci USA 2009;106:15049–54.
Salome R, Kremer Y, Dieudonne S, Leger JF, Krichevsky O, Wyart C, et al. Ultrafast random-access scanning in two-photon microscopy using acousto-optic deflectors. J Neurosci Methods 2006;154:161–74.
Stosiek C, Garaschuk O, Holthoff K, Konnerth A. In vivo two-photon calcium imaging of neuronal networks. Proc Natl Acad Sci USA 2003;100:7319–24.
Stuurman N, Amodaj N, Vale RD. Micro-Manager: open source software for light microscope imaging. Microsc Today 2007;15:42–3.
Suter BA, O'Connor T, Iyer V, Petreanu LT, Hooks BM, Kiritani T, et al. Ephus: multipurpose data acquisition software for neuroscience experiments. Front Neural Circuits 2010;4:100.
Tamamaki N, Yanagawa Y, Tomioka R, Miyazaki J, Obata K, Kaneko T. Green fluorescent protein expression and colocalization with calretinin, parvalbumin, and somatostatin in the GAD67-GFP knock-in mouse. J Comp Neurol 2003;467:60–79.
Wilt BA, Burns LD, Wei Ho ET, Ghosh KK, Mukamel EA, Schnitzer MJ. Advances in light microscopy for neuroscience. Annu Rev Neurosci 2009;32:435–506.